A short manual for 2d FLI - two-dimensional Fast Laplace Inversion

DRAFT

Schlumberger-Doll Research
1 Hampshire Street
Cambridge, MA 02139

March 20, 2007

2D FLI is an algorithm to perform two-dimensional numerical Laplace inversion, developed at Schlumberger-Doll Research. This algorithm solves the integral equation:

$$M(\tau 1,\tau 2) = \iint dT1 dT2 \cdot F(T1,T2)\exp(-\tau 1/T1)\exp(-\tau 2/T2) + Noise$$

where M is the signal as a function of $\tau 1$ and τ 2 and the algorithm solves for F(T1,T2) subject to the non-negativity constraint, F >= 0. This program also provides an algorithm for the 1D Laplace inversion. The corresponding equation is:

$$M(\tau 2) = \int dT2 \cdot F(T2)\exp(-\tau 2/T2) + Noise$$

The data here is 1-dimensional in the sense that M is a function of one parameter, tau2. The algorithm is not limited to the exponential kernel functions as in above equations.

Details were published in the following papers:

1. L. Venkataramanan et. al., IEEE Tran. SP. 50, 1017-1026 (May, 2002).
2. Y.-Q. Song, et al., J. Magn. Reson. 154, 261-268(2002).
3. M D Hurlimann and L Venkataramanan, J. Magn. Reson. 157, 31 (2002).
4. US Patent 6,462,542 B1, October 8, 2002

This document describes the use of the 1d and 2d FLI program, FLI 1.0.4, released on August 30, 2006.

How to use the program

This version of the program is different from the earlier versions in that the inversion is performed with two main functions: FLI1d and FLI2d. They follow the matlab function convention with well-defined inputs and output parameters. The intermediate results are held by the temporary parameters within the functions and not available as outputs.

As one of the input parameters, the Kernel matrix is supplied to the FLI1d or FLI2d. Thus, this program is not limited to the exponential kernels (such as in Laplace inversion) and it can be used for other inversions with a different kernel.

For example, to create the kernel for exponential decay, one may use the following matlab routine:

```
Tau = logspace(-3,1,30)';
T1 = logspace(-3,1,100);
Kernel_1 = inline('exp(- Tau * (1./ TimeConst))','Tau','TimeConst');
K_1 = Kernel_1 (Tau,T1);
```

The program FLI1d and FLI2d will perform the singular value decomposition of the kernel matrices if necessary. However, repeated execution of SVD can be avoided by providing the decomposition as inputs, i.e. S, V, and U.

The program allows several different input of the regularization parameter as inAlpha in FLI1d, for example:

```
function [FEst, parameter, Fitdata] = FLI1d(inputData,inKernel,inAlpha,U,S,V)
```

The choice of inAlpha is:

```
 0:                  BRD method
positive constant:   fixed regularization parameter
-1:                  full span, then a heel is determined
-2:                  t1heel method, a dynamic and quick method to find the heel
```

The resulting alpha parameter is output in parameter.

Requirement:

Matlab5.3 and above,
optimization toolbox

Files in the distribution (FLIv1.0.4):
Program files: (.m files are the header, and .p are compiled code)

FLI1d.m
FLI1d.p
FLI2d.m
FLI2d.p
FLIEstimate.m
FLIEstimate.p
FLIEstimate1d.m
FLIEstimate1d.p
FLIminfun.m
FLIminfun.p

Examples and plots:

FLIPlot2dDT2.m
FLIPlot2dT1T2.m

T1T2analysis.m

testFLI1d-result.fig
testFLI1d.m
testFLI1dresult.mat
testFLI2d-result.fig
testFLI2d.m
testFLI2dresult.mat

release notes
ReadME.pdf (this file)

Main files:

FLI2d

This is the main program for the 2D inversion algorithm. Function prototype:

```
function [FEst, parameter, Fitdata] =
FLI2d(inputData,inKernel1,inKernel2,inAlpha,U1,S1,V1,U2,S2,V2)
```

FLI1d

This is the main program for the 1D inversion algorithm. Function prototype:

```
function [FEst, parameter, Fitdata] = FLI1d(inputData,inKernel,inAlpha,U,S,V)
```

FLIEstimate.m

Core routine to optimize for FEst for 2D inversion.

FLIminfun.m

Minimization function, used in FLIEstimate.m and internally by matlab.

FLIEstimate1d.m

Core routine to optimize for FEst specific to 1D inversion.

Input data and parameters

1D INVERSION:

```
function [FEst, parameter, Fitdata] = FLI1d(inputData,inKernel,inAlpha,U,S,V)
%
%   Fast Laplace inversion for 1d data
%   function [FEst, parameter, Fitdata] = FLI1d(inData,inKernel,inAlpha)
%   function [FEst, parameter, Fitdata] = FLI1d(inData,inKernel,inAlpha,U,S,V)
%
%    inData : data vector
%    inKernel: kernel matrix
%    inAlpha: 0:BRD method, positive constant: fixed reg, -1:full span, -2:t1heel method.
%    U,S,V: K=U*S*V', if they are provided, then SVD of K will be skipped.
```

2D INVERSION:

```
function [FEst, parameter, Fitdata] =
FLI2d(inputData,inKernel1,inKernel2,inAlpha,U1,S1,V1,U2,S2,V2)
%   Two-dimensional inversion of Folh integral, including Laplace
%   inversion.
%   Syntax:
%   [FEst, parameter, Fitdata] = FLI2d(inData,inKernel1,inKernel2,inAlpha,U1,S1,V1,U2,S2,V2)
```

```
%   [FEst, parameter, Fitdata] = FLI2d(inData,inKernel1,inKernel2,inAlpha)
%
%    inData : data matrix
%    inKernel: kernel matrices
%    inAlpha: 0:BRD method, positive constant: fixed reg, -1:full span, -2:t1heel method.
%    U,S,V: K=U*S*V', if they are provided, then SVD of K will be skipped.
```

Execution of the program

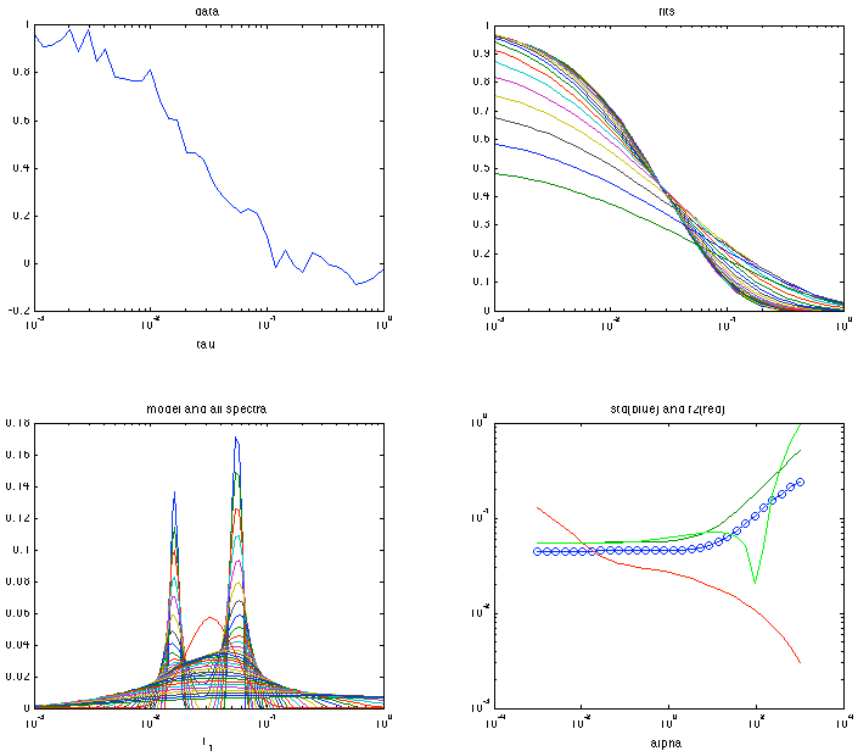
In matlab, once the path is set to include the FLI files, command FLI1d or FLI2d will execute the program.

The result of the inversion is stored in a file whose name is the input file name plus “Re.mat”.

Examples

Example 1:
File: testFLI1d.m, testFLI1dresults.mat, testFLI1d-result.fig

Inversion of 1D data. Upper left panel: the original data; upper right: fits; lower left: spectra at different regularization parameter; lower right: error analysis

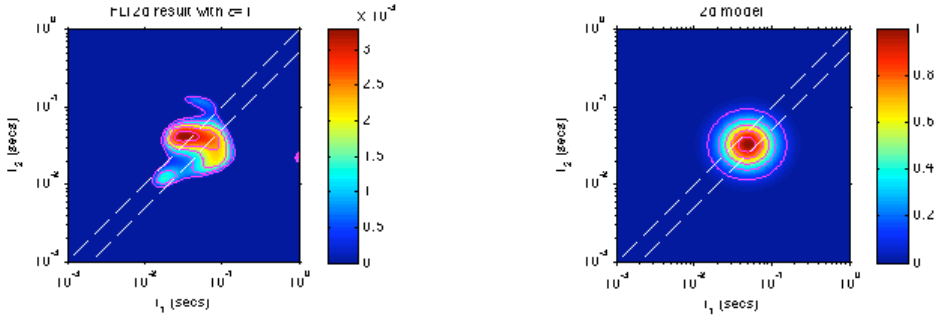


Example 2:

Inversion of 2D data.

Files: testFLI2d.m, testFLI2d-result.fig, testFLI2dresult.mat

Figure: left: inversion result; right: model spectrum.



User Modifications of FLI

This section refers to the earlier version. Some of it might still be applicable for the current version. Will be updated in the future.

Modification to FLI.m

1. Modification for different regularization parameter and schemes

Regularization (alpha) can affect the final inversion result and its value depends on the signal-to-noise ratio of the data. For example, a small alpha can be used with data of high S/N to obtain details of the spectrum. However, when S/N is low, a larger alpha should be used often resulting in a smoother spectrum. In practice, one should perform the inversion with several values of alpha in order to gain confidence in the features observed in the inversion results.

The regularization is set by the following parameter in FLI1d and FLI2d:

Alpha_Auto = 0:	fixed regularization, and the value is AlphaStart.
Alpha_Auto = 1:	automatic regularization using BRD method. For this method to work well, the Data should be free from systematic errors and NoiseStd reflect the variance of the random noise.
Alpha_Auto=2:	A series of regularization will be used to calculate the inversion. Then the heel of the fit error (c2) will be found to be the optimal regularization.

2. Modifications for different range of T1 and T2

These following variables should be set to reflect the range of T1 and T2 for the specific data set for optimal inversion result. For example, the minimum T1 (InitTime_T1) should not be smaller than the smallest Tau_1 and the maximum T1 (FinalTime_T1) should not be much larger than the maximum Tau_1. The similar rule should apply to T2.

The number of points for T1 and T2 should not be much smaller than 100. For example, Number_T1=20 might be too small to allow a good fit for some data.

The ranges of T1 and T2 are set by the following in FLI.m:

```
InitTime_T1 = .01;          % The initial value of T1 (in seconds)
FinalTime_T1 = 10;          % The final value of T1 (in seconds)
Number_T1 = 100;            % The number of T1's
```

```
InitTime_T2 = .001;          % The initial value of T2 (in seconds)
FinalTime_T2 = 10;           % The final value of T2(in seconds)
Number_T2 = 100;             % The number of T2's
```

3. Modifications for DC offset

The non-ideal inversion of the 180 degree pulse and the rf inhomogeneity would result in a DC offset. This offset can be estimated by adding a column in K_1. This feature is enabled by:

```
AllowDCOffset = 1;   % allow a dc offset
% AllowDCOffset = 0;         % no offset
```

4. Modifications for different kernels

The inversion kernels can be modified to suit the specific data. The kernels are defined as inline functions and calculated later.

```
%exponential decay
%Kernel_1 = inline('exp(- Tau * (1./ TimeConst))','Tau','TimeConst');

%inv recovery
Kernel_1 = inline('1- 2*exp( - Tau * (1./ TimeConst))','Tau','TimeConst');
```

5. Modification to neglect the non-negativity constraint

It is not implemented in the current software.

The principle of the modification is discussed in the reference 1 (equation 36).

Modification to FLIestimate1d.m, FLIestimate.m, FLIminfun.m, etc

User should not modify these files.

Modification to plotting files and other examples

The files in those directories are provided as examples. Users may use them as is or as templates for their own routines to display the inversion results.

Contact information

Y.-Q. Song, ysong@slb.com
L. Venkataramanan, lvenkataramanan@ridgefield.oilfield.slb.com
M. D. Hurlimann, mhurlimann@ridgefield.oilfield.slb.com

Schlumberger-Doll Research, 1 Hampshire Street, Cambridge, MA 02139